Path Optimization for Humanoid Walk Planning: an Efficient Approach

Antonio El Khoury, Michel Taïx, Florent Lamiraux

PATH OPTIMIZATION FOR HUMANOID WALK PLANNING: AN EFFICIENT APPROACH

Antonio El Khoury[1,2], Michel Taïx[1,2], Florent Lamiraux[1,2]
[1]CNRS ; LAAS ; 7 avenue du colonel Roche, F-31077 Toulouse Cedex 4, France
[2]Université de Toulouse ; UPS, INSA, INP, ISAE ; UT1, UTM, LAAS ; F-31077 Toulouse Cedex 4, France
aelkhour@laas.fr, taix@laas.fr, florent@laas.fr

Keywords: humanoid robot, motion planning, walk planning, nonholonomic, holonomic, optimization, A*, HRP-2, sampling.

Abstract: The paper deals with walk planning for humanoid robots in a cluttered environment. It presents a heuristic and efficient optimization method that takes as input a path computed for the robot bounding box by a random diffusion algorithm, and produces a path where a discrete set of configurations has been reoriented using an A* search algorithm. The resulting whole-body trajectory is a realistic and time-optimal one. We validate our method in various virtual scenarios and on the real humanoid robot HRP-2.

1 RELATED WORK AND CONTRIBUTION

The problem of humanoid walk planning can be defined as follows: given an environment and a humanoid robot with start and goal placements, we need to find a whole-body trajectory that is collision-free. This trajectory should ideally represent realistic human motion, i.e. a human being would have walked the same way and taken a similar path if put in the same conditions. This additional constraint is desirable since humanoid robots are bound to move around in human-made environments such as homes, offices, and factories, and because it can help them blend in between humans.

Finding such trajectory has been extensively studied and is still a current field of research.

Figure 1: Humanoid Robot HRP-2 using holonomic motion (side-stepping) to pass between two chairs.

1.1 Humanoid Walk Planning

The problem of motion planning is now well formalized in robotics, and several books present the various approaches (Latombe, 1991; Choset et al., 2005; LaValle, 2006). Deterministic algorithms have the nice property of being complete, i.e. they can tell whether a solution exists or not. These algorithms are however limited for simple systems because the computational cost increases exponentially with the number of degrees of freedom (DoF). They cannot thus be applied for human-like figures which are high-DoF systems.

With sampling-based methods, which rely on random sampling in the configuration space (CS) and use for instance Probabilistic Roadmaps (PRM) (Kavraki et al., 1996) or Rapidly-Expanding Random Trees (RRT) (Kuffner and LaValle, 2000), it is possible to solve problems for systems with a large number of DoF.

Such algorithms possess the weaker propery of probabilistic completeness, meaning that if a solution exists, it will be surely found in finite time. In practice, probabilistic algorithms such as RRT-Connect can compute feasible trajectories for high-DoF devices in a reasonable time.

The motion planning problem is indeeed a complex one in the case of humanoid robots, which are highly sized redundant systems that have to verify stabiity constraints. Different planning strategies can be found in litterature.

One category relies on whole-body task planning: kinematic redundancy is used to accomplish tasks with different orders of priorities (Khatib et al., 2004; Kanoun et al., 2009; Dalibard et al., 2009). Static balance and obstacle avoidance can thus be defined as tasks that the algorithm has to respect.

One algorithm was specifically designed for humanoid footstep planning in (Kuffner and LaValle, 2000). Starting from an initial footstep placement, it uses an A* graph search algorithm (Hart et al., 1968) to explore a discrete set of footstep transitions. The search stops when the neighborhood of the goal footstep placement is reached. This approach is not practical in some special environments with narrow passages, and notable improvement would be the work of (Xia et al., 2009) which reduced the computational cost of footstep planning by using an RRT-style planning algorithm.

Another strategy consists of dividing one high-dimensional problem into smaller problems and solving them successively (Zhang et al., 2009). The idea of dividing the problem into a two-stage approach is described in (Yoshida et al., 2008): A 36-DoF humanoid robot is reduced to a 3-DoF bounding box linked to a 6-DoF bar that the robot is holding. If we consider the robot alone, a configuration q consists of 2 translation variables x and y in the horizontal plane and 1 rotation variable θ around the vertical axis. Using the robot simplified model, the PRM algorithm first solves the path planning problem and generates a feasible path for the bounding box. A geometric decomposition of the path places footsteps on it, and a walk pattern generator based on (Kajita et al., 2003) finally produces the whole-body trajectory for the robot. During this second stage, dynamic constraints are taken into account for some specific aspects of humanoid planning: static and dynamic balance, for instance require robot Center of Mass (CoM) and the Zero-Momentum Point (ZMP) respectively to lie within its support polygon. In (Moulard et al.,), this two-stage approach is also used; numerical optimization of the bounding box path path produces a time-optimal trajectory that is constrained by foot speed and distance to obstacles.

Another important issue is the notion of holonomic motion: while wheeled robots always remain tangent to their path, thus following a nonholonomic constraint, legged robots can also move sideways, and their motion can be described as holonomic. The path planning scheme in (Yoshida et al., 2008) is designed to this end; a PRM algorithm first builds a roadmap with Dubins curves (Dubins, 1957); but such curves impose a nonholonomic constraint and narrow passages cannot be crossed. The roadmap is therefore enriched in a second step with linear local paths that allow the robot to move sideways. As a result this planning scheme generates motion such that the robot remains tangent to its path most of the time and uses sidestepping only in narrow passages.

Moreover, (Mombaur et al., 2010; Truong et al., 2010) conducted a series of walking experiments that allowed them to build a model for human gait; if a person walks long distances, its body will roughly remain tangent to its path most of the time, while holonomic motion will be used over smaller distances. This is a property that is desirable for our paths if we want to have realistic motion, and holonomic motion can be as well used to pass through narrow spaces. This work shows that it is important to decouple the trajectory orientation and the robot orientation in order to produce realistic walking.

1.2 Contribution

The work of (Moulard et al.,) solves the walk planning problem in a natural way, i.e it uses numerical optimization to minimize the time it takes the robot to walk along the path while following speed and obstacle distance constraints. After having tried this approach, we came to the empirical conclusion that achieving successful numerical optimization in any kind of environment is a difficult and computationally expensive task; indeed, it requires computing a large set of parameters to fully define the optimized path.

While using the same two-stage approach of (Yoshida et al., 2008), we propose a simpler heuristic method that generates realistic time-optimal humanoid trajectories. We first replace the PRM algorithm and the Dubins local paths with an RRT-Connect algorithm and linear local paths. The path is then optimized by locally reorienting the robot bounding box on a discrete set of configurations of the path. We give priority to nonholonomic motion and use holonomic motion such as side-stepping only to pass in narrow passages and avoid nearby obstacles.

Next section presents our method and explains how it is integrated in the motion planning scheme.

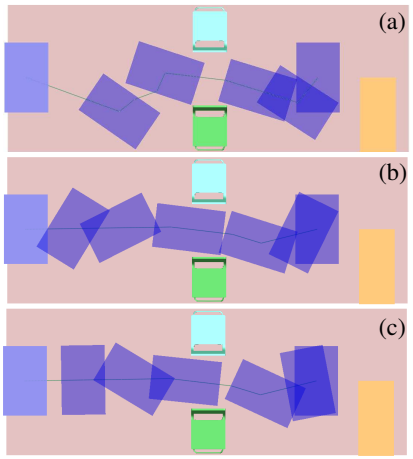

Figure 2: Top view: (a) RRT-Connect path for the bounding box passing between two chairs. (b) Optimized bounding box path by random optimization. (c) Optimized bounding box after adding optimization by regular sampling.

We show in section 3 some examples on different scenarios. The HRP-2 platform (Kaneko et al., 2004) is used to execute a real scenario.

2 OPTIMIZATION BY REGULAR SAMPLING

Assuming we have full knowledge of the environment, the RRT algorithm produces in offline mode a collision-free piecewise linear path P_{RRT} for the robot bounding box, i.e. the path consists of the concatenation of linear local paths LP_{RRT}.

Due to the random nature of RRTs, P_{RRT} may not be optimal in terms of length, and a preliminary random shortcut optimization can be run in order to shorten it (See Figure 2). While the optimized path P is collision-free, the bounding box orientation is such that it could lead to an unrealistic trajectory that is, moreover, not time-optimal. For instance, the humanoid robot could spend a long time walking sideways or backwards over a long distance in an open space.

We use a decoupled approach and introduce an additional optimization stage to address this issue in the next section.

2.1 Bounding Box Path Optimization

Note that each configuration q can be written as:

$$q = (\mathbf{X},\ \theta) \tag{1}$$

where $\mathbf{X} = (x,\ y)$ is a vector describing the bounding box position in the horixzontal plane, and θ gives its orientation. Our optimizer reorients the bounding box along P by changing θ while keeping the same value for $\mathbf{X}$.

For this purpose, we run an A* search algorithm; we regularly sample P and, using a discrete set of possible orientations for each sample configuration and an adequate heuristic estimation function, we modify the bounding box orientation along P. An optimized path P_{opt} is created and leads to a realistic time-optimal trajectory for the humanoid robot. Each of our approach main components is described in the following sections.

2.1.1 Preliminaries

After running Random Optimization on the piecewise linear path P_{RRT}, the path P is also piecewise linear and can be written as a continuous map P from the interval $[0, L_P]$ to the three-dimensional configuration space (CS), where L_P is the length of P. Any configuration q of P at distance l from the start configuration can be then written as

$$q = P(l) \quad \text{if } l \in [0, L_P]. \tag{2}$$

We also have $q_s = P(0)$ and $q_g = P(L_P)$

Let $d_{sample} \in \mathbb{R}_+^*$ be a sampling distance. Sampling P with a distance d_{sample} means dividing each local path LP_j of length L_{LP_j} of P into smaller local paths of length d_{sample}; each new local path end is a sample configuration. The n^{th} sample configuration of P in its intial state can be obtained by indexing new local path ends starting from q_s, and is denoted by q_n^{init}.

Now let us define the possible orientation states. We aim to make a humanoid robot reach its goal as soon as possible. Since our robot is faster while walking straight than while side-stepping, we will attempt

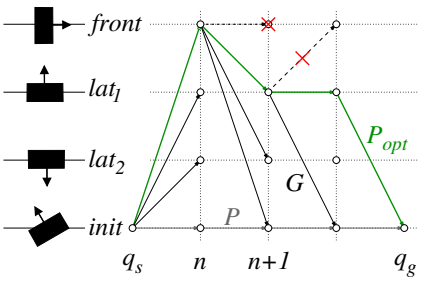

Figure 3: Each initial sample configuration can be rotated and be in one of four states. Starting from q_s, the A* search algorithm searches the graph G that contains only valid nodes and arcs to produce an optimized path P_{opt}.

to change the orientation of each inital sample configuration q_n^{init} such that the bounding box is tangent to the local path and introduce a new configuration denoted by q_n^{front}. To take into account the fact that there may be obstacles that forbid a frontal orientation, we also create $q_n^{lat_1}$ and $q_n^{lat_2}$ that are rotated by $\frac{\pi}{2}$ and $-\frac{\pi}{2}$ related to the local path tangent (See Figure 3). One particular case is local path end configurations: Figure 4 shows how we use the mean direction of the two adjacent local paths to define frontal and lateral configurations. This is done to ensure a smooth transition between two local paths.

A sample configuration whose orientation is unknown will be denoted by q_n^{state}. It can have any orientation state of the set $\{init, front, lat_1, lat_2\}$ except for q_s and q_g which remain in their initial state. Ideally, our algorithm should be able, as long there are no obstacles, to put each sample configuration in the frontal state, create a new path P_{opt}, and generate a time-optimal trajectory for the robot.

We run an A* search to achieve our goal, and we fully describe the algorithm functions in the following section.

2.1.2 A* Function Definition

An A* search algorithm can find an optimal path in a graph as long as a graph and an evaluation function are correctly defined. Starting from q_s, A* expands in each iteration the possible transitions from one sample to the next one in the graph and evaluates with the evalutation function the cost of going through each different state (See Figure 3).

A graph G is defined to be a set of nodes and arcs. Each node consists of a configuration q_n^{state} and is indexed by its sample number; all nodes that have the same index represent different orientation states for the bounding box at one point in the path, and are at equal depth in G if we consider q_s as its root. An arc represents a linear local path that connects exclusively one sample configuration to the next one and is denoted by $q_n^{state_n} q_{n+1}^{state_{n+1}}$. A valid node is defined

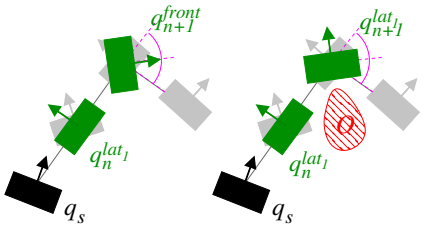

Figure 4: Local path end configurations are reoriented by taking into account the adjacent local path directions.

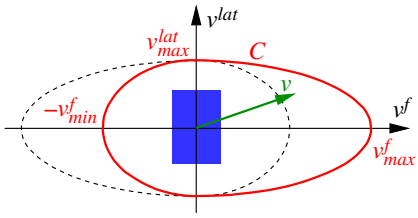

Figure 5: The bounding box speed vector v is bounded inside an area defined by the speed constraint C.

to be a configuration that does not generate collisions with obstacles, and a valid arc is a collision-free linear local path.

We could chose to build the whole graph G before running A* by testing all possible nodes and arcs and making sure they are collision-free. But collision tests are rather slow, and A* uses a heuristic estimation function to avoid going through all nodes. In order to save computation time, we start with an empty graph G and build the necessary nodes and arcs only when necessary. We need to define a successor operator for this puropose.

The Successor operator $\Gamma(q_n^{state_n})$ Its value for any sample node $q_n^{state_n}$ is a set of pairs $\{(q_{n+1}^{state_{n+1}}, c_{n,n+1})\}$, where $q_{n+1}^{state_{n+1}}$ denotes a successor node, and $c_{n,n+1}$ is the cost of going from $q_n^{state_n}$ to $q_{n+1}^{state_{n+1}}$.

A node $q_n^{state_n}$ can thus have up to four successors. Note that collision tests are run for each node and only valid nodes and arcs are added to the graph G. Arcs $q_n^{lat_1} q_{n+1}^{lat_2}$ and $q_n^{lat_1} q_{n+1}^{lat_2}$ are never added because switching from a lateral orientation to another is never a better alternative to keeping the same lateral orientation state.

We define the cost $c_{n,n+1}$ to be the distance $D(q_n^{state_n}, q_{n+1}^{state_{n+1}})$ between two nodes of G; it computes the walk time from $q_n^{state_n}$ to $q_{n+1}^{state_{n+1}}$. D should give short time for frontal walk and penalize sidestepping and walking backwards. We therefore define the speed constraint C to be

$$C = \begin{cases} (\frac{v^f}{v_{max}^f})^2 + (\frac{v^{lat}}{v_{max}^{lat}})^2 - 1 & \text{if } v^f >= 0 \\ (\frac{v^f}{v_{min}^f})^2 + (\frac{v^{lat}}{v_{max}^{lat}})^2 - 1 & \text{if } v^f < 0 \end{cases} \quad (3)$$

where v^f and v^{lat} are respectively the frontal and lateral speed, and v_{min}^f v_{max}^f and v_{max}^{lat} their minimum and maximum values.

Figure 5 shows that the inequality $C \leq 0$ constrains the bounding box speed in a zone defined by

two half-ellipsoids. $D(q_n^{state_n}, q_{n+1}^{state_{n+1}})$ can be then computed by sampling the arc and integrating this speed constraint along it.

Having fully definined the successor operator, we can now define the A* evaluation function that will allow it to choose wich node to expand at each iteration.

The Evaluation Function $f(q_n^{state})$ It is the actual cost of an optimal path going through q_n^{state} from q_s to q_g and can be written as:

$$f(q_n^{state}) = g(q_n^{state}) + h(q_n^{state}) \tag{4}$$

where $g(q_n^{state})$ is the actual cost of the optimal path from q_s to q_n^{state} and $h(q_n^{state})$ is the actual cost of the optimal path from q_n^{state} to q_g.

During execution of A*, we usually only have an estimate $\hat{g}(q_n^{state})$ of $g(q_n^{state})$ and we need to define an estimate heuristic function $\hat{h}(q_n^{state})$ such that $\hat{h}(q_n^{state}) \leq h(q_n^{state})$ to ensure that the algorithm is admissible, i.e. the path from q_s to q_g is optimal. Since the robot is fastest while walking straight forward in the absence of obstacles, we define $\hat{h}(q_n^{state})$ as:

$$\begin{aligned}\hat{h}(q_n^{state}) &= D(q_n^{state}, q_{n+1}^{front}) \\ &+ \sum_{k=1}^{N_{sample}-n-2} D(q_{n+k}^{front}, q_{n+k+1}^{front}) \\ &+ D(q_{n+1}^{front}, q_g)\end{aligned} \tag{5}$$

where N_{sample} is the total number of initial sample configurations in P including q_s and q_g, and the three distance terms respectively represent the cost of moving towards the frontal orientation, of walking frontally along P, and of the final motion towards the goal node.

Now that our A* functions are fully defined, we can, as described in Algorithm 1, run a search algorithm to compute an optimal path P_{opt} by changing the orientation of each sample node. An example is shown in Figure 6.

Algorithm 1 Regular-Sampling-Optimization (P)

$P_{opt} \leftarrow \emptyset$
$q_s \leftarrow$ start configuration of P
$q_g \leftarrow$ goal configuration of P
$d_{sample} \leftarrow$ sampling interval size
$P_{opt} \leftarrow$ A*(q_s, q_g, P, d_{sample})
return P_{opt}

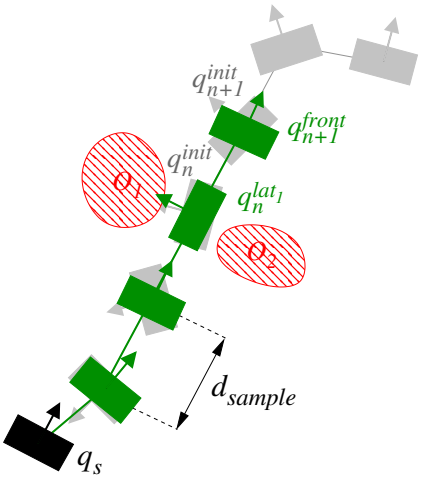

Figure 6: Local paths are regularly sampled (grey) and each sample configuration is reoriented (green) while considering obstacles (red).

2.2 Motion Generation for a Humanoid Robot

A collision-free path P for the robot 3-DoF bounding box can be found using RRT-Connect and Random Optimization. One should note that this optimization tends to push the bounding box close to obstacles if we use a regular euclidean distance. This may prevent our optimizer from sucessfully putting configurations in a frontal state, and we therefore use the same distance function defined in subsubsection 2.1.2 to penalize lateral and backwards walking.

Our regular sampling optimization is then applied on the path and produces a path P_{opt} that gives priority to nonholonomic motion.

We now have the bounding box trajectory and we would like to make the humanoid robot walk along this trajectory . A footstep sequence is thus generated along P_{opt} by geometric decomposition of the path, and the pattern generator cited in subsection 1.1 then produces the robot whole-body trajectory by computing a full configuration every 5ms. Since the robot remains inside the bounding box while walking and the pattern generator avoids self-collisions, we are guaranteed to get a collision-free trajectory. The trajectory is stored in a file that can be played on the humanoid robot in open loop.

3 EXAMPLES

This section presents experimental results of our path optimizer after it has been inserted in the previously

Table 1: Computational time (ms) of each planning stage for the presented scenarios.

	RRT-Connect	RO	RSO	Robot Trajectory	Total
Chairs	3,968	1,887	2,144	66,140	74,140
Galton	91.69	2,497	237.8	65,730	68,560
Apartment	1,212	2,425	2,412	222,800	228,800

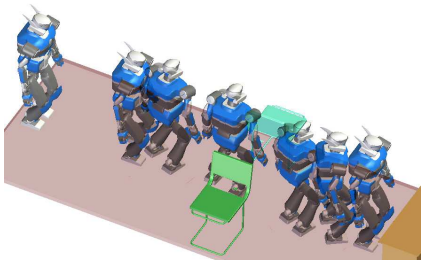

Figure 7: Perspective view of the simulated HRP-2 trajectory on the final optimized path passing between two chairs.

described walk planning scheme.

We ran simulations of the humanoid robot HRP-2 in three scenarios. The first one is a small environment where HRP-2 has to cross a distance of about 4 m while passing between two chairs. The second environment is uncluttered with few obstacles laying around, while the last one is a bigger apartment environment with 3 rooms where the robot has to move from one room to another while passing through doors. Due to its shoulder wideness and swaying motion, HRP-2 cannot walk frontally through a normal door. We therefore use doors that are 1.3 m wide, such that the shoulder length vs door wideness ratio is equal to that of a human passing through a normal door. We also replay on the real robot HRP-2 the trajectory that is computed offline for the chairs scenario.

We set our distance parameters v^f_{max}, v^{lat}_{max}, v^f_{min} to 0.5, 0.1, and 0.25 respectively. We also need to define the sampling interval size parameter d_{sample} for our optimizer. Preliminary tests showed that a value equal to $\frac{size}{6}$, where *size* is our humanoid's size, gave satisfying results.

The implementation of our algorithm uses KineoWorks™ (Laumond, 2006) implementation of random diffusion algorithms and collision checking. Simulations were performed on a 2.13 GHz Intel Core 2 Duo PC with 2 GB RAM.

Table 1 shows computationt times for each stage of our planning scheme: RRT-Connect, Random Optimization (RO), our optimizer which we call Regular Sampling Optimization (RSO), and the whole-body robot trajectory generation.

In order to show our optimizer contribution, we also measure robot walk times by creating a trajectory straight away after RO, and comparing it with a trajectory where we added the RSO. Walk times can be seen in Table 2.

Table 2: Humanoid robot walk time (s) for the presented scenarios using RO alone and a RO-RSO combination.

	RO	RO+RSO
Chairs	40	35
Galton	66	57
Apartment	200	120

3.1 "Chairs" Scenario

Figure 2 shows the bounding box RRT path and the RO path for the chairs scenario. We can see that RO creates a shorter path, but the bounding box starts rotating from the beginning of the path even though the two chairs are still far. This causes the robot trajectory to be unrealistic on one hand and, since walking sideways takes a longer time than walking straight, to be not time-optimal on the other hand.

But after applying our RSO, we see that the bounding box stays frontal and rotates only when it reaches the chairs. Figure 7 and Table 2 show that the walk time shorter by 5 s and the final trajectory for HRP-2 is more realistic. Note that the RSO takes 2,144 ms to be executed on the chairs path, which is less than 3% of the total walk planning computation time.

The same trajectory was executed in open loop

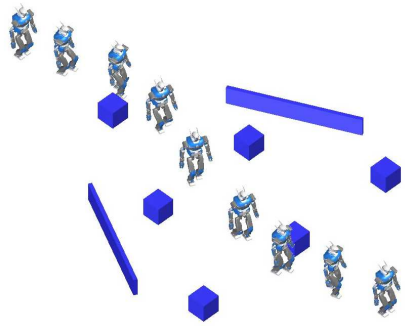

Figure 8: Perspective view of HRP-2 optimized trajectory in the Galton board scenario.

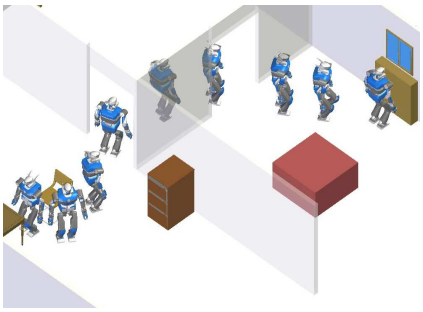

Figure 9: Perspective view of HRP-2 optimized trajectory in the apartment scenario.

on the real humanoid robot HRP-2, and it suceeded in prioritizing nonholnonomic motion while walking sideways to pass between the chairs (See Figure 1).

3.2 "Galton" Scenario

Since this is an uncluttered environment, HRP-2 should be able to walk straight on the whole trajectory. We see in Figure 8 that this is the case; indeed RO uses the speed constraint defined in subsubsection 2.1.2 to shorten the RRT path while keeping sufficiently away from obstacles.

3.3 "Apartment" Scenario

Our planning scheme is finally applied in the apartment scenario. In Figure 9, we see that HRP-2 manages to walk frontally through the doors. As with the previous scenarios, the final trajectory is more realistic than a trajectory where RSO is not used. The added computation time for using RSO is 2,412 ms, which is insignificant compared to the 228 s it takes for the whole planning scheme.

Furthermore, since the environment is significantly larger and more constrained than the previous ones, the walk time difference is more striking: Table 2 shows that it takes the robot 80 s less to cross the apartment when an RO-RSO combination is applied to the RRT path. We realized a video showing the superimposed trajectories and the time difference. It can be viewed along with videos showing each planning stage for all scenarios at `http://humanoid-walk-planning.blogspot.com/`

4 CONCLUSION

In this paper, we have presented a novel simple optimization method for humanoid robot walk planning. It uses an A* search that takes as input a path computed for the robot bounding box by a random diffusion algorithm, and produces a path where a discrete set of configurations has been reoriented to generate a realistic time-optimal humanoid trajectory. The results show that new trajectories are much more satisfying while the added computation time is insignificant compared to the whole planning scheme.

Certainly, the decoupling between trajectory orientation and robot orientation of this work can be used in other fields such as graphics animation on digital actors to adapt the body orientation with respect to the goal during locomotion. With a motion capture library containing pre-recorded nonholonomic and holonomic walk behaviors, it is possible to lay this behavior on the actor trajectory and produce realistic movements.

REFERENCES

Choset, H., Lynch, K. M., Hutchinson, S., Kantor, G. A., Burgard, W., Kavraki, L. E., and Thrun, S. (2005). *Principles of Robot Motion: Theory, Algorithms, and Implementations*. MIT Press, Cambridge, MA.

Dalibard, S., Nakhaei, A., Lamiraux, F., and Laumond, J.-P. (2009). Whole-body task planning for a humanoid robot: a way to integrate collision avoidance. In *Humanoid Robots, 2009. Humanoids 2009. 9th IEEE-RAS International Conference on*, pages 355 –360.

Dubins, L. E. (1957). On curves of minimal length with a constraint on average curvature, and with prescribed initial and terminal positions and tangents. *American Journal of Mathematics*, 79(3):pp. 497–516.

Hart, P., Nilsson, N., and Raphael, B. (1968). A Formal Basis for the Heuristic Determination of Minimum Cost Paths. *IEEE Transactions on Systems Science and Cybernetics*, 4(2):100–107.

Kajita, S., Kanehiro, F., Kaneko, K., Fujiwara, K., Harada, K., Yokoi, K., and Hirukawa, H. (2003). Biped walking pattern generation by using preview control of zero-moment point. In *Robotics and Automation, 2003. Proceedings. ICRA '03. IEEE International Conference on*, volume 2, pages 1620 – 1626 vol.2.

Kaneko, K., Kanehiro, F., Kajita, S., Hirukawa, H., Kawasaki, T., Hirata, M., Akachi, K., and Isozumi, T. (2004). Humanoid robot hrp-2. In *Robotics and Automation, 2004. Proceedings. ICRA '04. 2004 IEEE International Conference on*.

Kanoun, O., Yoshida, E., and Laumond, J.-P. (2009). An optimization formulation for footsteps planning. In *Humanoid Robots, 2009. Humanoids 2009. 9th IEEE-RAS International Conference on*, pages 202 –207.

Kavraki, L., Svestka, P., Latombe, J.-C., and Overmars, M. (1996). Probabilistic roadmaps for path planning in high-dimensional configuration spaces. *Robotics and Automation, IEEE Transactions on*, 12(4):566 –580.

Khatib, O., Sentis, L., Park, J., and Warren, J. (2004). Whole-body dynamic behavior and control of human-like robots. *I. J. Humanoid Robotics*, 1(1):29–43.

Kuffner, J.J., J. and LaValle, S. (2000). Rrt-connect: An efficient approach to single-query path planning. In *Robotics and Automation, 2000. Proceedings. ICRA '00. IEEE International Conference on*.

Latombe, J.-C. (1991). *Robot Motion Planning*. Kluwer Academic Publishers, Norwell, MA, USA.

Laumond, J.-P. (2006). Kineo cam: a success story of motion planning algorithms. *Robotics Automation Magazine, IEEE*, 13(2):90 –93.

LaValle, S. M. (2006). *Planning Algorithms*. Cambridge University Press, Cambridge, U.K. Available at http://planning.cs.uiuc.edu/.

Mombaur, K., Truong, A., and Laumond, J.-P. (2010). From human to humanoid locomotion–an inverse optimal control approach. *Auton. Robots*, 28:369–383.

Moulard, T., Lamiraux, F., and Wieber, P.-B. Collision-free walk planning for humanoid robots using numerical optimization. Retrieved from http://hal.archives-ouvertes.fr/hal-00486997/en/.

Truong, T.-V.-A., Flavigne, D., Pettre, J., Mombaur, K., and Laumond, J.-P. (2010). Reactive synthesizing of human locomotion combining nonholonomic and holonomic behaviors. In *Biomedical Robotics and Biomechatronics (BioRob), 2010 3rd IEEE RAS and EMBS International Conference on*, pages 632 –637.

Xia, Z., Chen, G., Xiong, J., Zhao, Q., and Chen, K. (2009). A random sampling-based approach to goal-directed footstep planning for humanoid robots. In *Advanced Intelligent Mechatronics, 2009. AIM 2009. IEEE/ASME International Conference on*, pages 168 –173.

Yoshida, E., Esteves, C., Belousov, I., Laumond, J.-P., Sakaguchi, T., and Yokoi, K. (2008). Planning 3-d collision-free dynamic robotic motion through iterative reshaping. *Robotics, IEEE Transactions on*, 24(5):1186 –1198.

Zhang, L., Pan, J., and Manocha, D. (2009). Motion planning of human-like robots using constrained coordination. In *Humanoid Robots, 2009. Humanoids 2009. 9th IEEE-RAS International Conference on*, pages 188 –195.